UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/449,782	11/26/1999	JAMES MCKEETH	MICS:0194	6698

37106 7590 03/23/2007

FLETCHER YODER P.C.
7915 FM 1960 RD. WEST
SUITE 330
HOUSTON, TX 77070

EXAMINER	
STEELMAN, MARY J	
ART UNIT	PAPER NUMBER
2191	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	03/23/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	09/449,782	MCKEETH, JAMES	
	Examiner	Art Unit	
	Mary J. Steelman	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *18 December 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-21 and 23-25* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-21 and 23-25* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☑ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This Office Action is in response to Remarks received 12/18/2006. Claims 1-21 and 23-25 are pending.

Response to Arguments

2. Applicant has argued, in substance, the following:

(A) In reference to the 35 U.S.C. 112 second paragraph rejection, as noted on page 7 of Remarks, the term 'command line utility' is consistent with and does not contradict definitions provided by Examiner. Additionally Applicant points to the Specification, page 4 for support for the limitation.

Examiner's Response:

The Specification, page 4, recites:

By way of example, consider a situation in which an executing application needs information of the type provided by command line utility CMD-UTIL, where CMD-UTIL represents any utility executable from a command line prompt (e.g., the 'dir' directory command of a Microsoft WINDOWS operating system or the 'head' command of an UNIX operating system). In accordance with the invention, the application invokes a system call of the form:

CMD-UTIL [PARAM] | REDIRECT ID..."

Examiner asks whether the 'executing application' is a command prompt terminal, screen such as can be found in a Microsoft environment? Does this terminal screen application invoke a command line utility? If so, Buxton, discloses a user interface / command line interpreter (col.

8: 45-47) and components that register themselves in registry with WIN 32 APIs (col. 9: 5-6). Col. 10: 1-4, "In addition to component contents 302...a number of entries in the system registry 250, in the form of registry keys or subkeys, are associated with each component." Qureshi provides more specific registration details by disclosing, (col. 3: 26-46) "a registration system for registering configuration information for an application program...The application program has a setup program (invoke a call of a command line utility) for storing configuration information in the registry during registration...passing to the registration routine a configuration file that contains a description of the configuration information to be loaded into the registry for the application program... The invoked registration routine opens the indicated configuration file and retrieves the description of the configuration information to be loaded. The registration routine (output from the utility is stored in the registry / system storage at a location identified by the identifier / key) then stores the configuration information in the registry..." Output is 'redirected' to the registry.

(B) Applicant has argued (page 9, 1st full paragraph), "These registration routines are not command line utilities.

Examiner' Response: Examiner disagrees. As noted above, Buxton, discloses a user interface / command line interpreter (col. 8: 45-47) and components that register themselves in registry with WIN 32 APIs (col. 9: 5-6). Col. 10: 1-4, "In addition to component contents 302...a number of entries in the system registry 250, in the form of registry keys or subkeys, are associated with each component." Qureshi provides more specific registration details by disclosing, (col. 3: 26-

46) "a registration system (utilities) for registering configuration information for an application program...The application program has a setup program (invoke a call of a command line utility) for storing configuration information.

(C) As noted on page 8, bottom paragraph, "There is absolutely no suggestion anywhere in Buxton that the OLE libraries are able to call WIN 32 APIs to modify the operating system registry are executable from a command line prompt."

Examiner's Response:

Examiner disagrees. As noted above, Buxton, discloses a user interface implemented with a command line interpreter (user interface at command line prompt) (col. 8: 45-47) and components that register themselves in registry with WIN 32 APIs (col. 9: 5-6). Col. 8: 45, "enables a user to interact" (are executable) Col. 10: 1-4, "In addition to component contents 302...a number of entries in the system registry 250, in the form of registry keys or subkeys, are associated with each component." Qureshi provides more specific registration details by disclosing, (col. 3: 26-46) "a registration system (utilities) for registering configuration information for an application program...The application program has a setup program (invoke a call of a command line utility) for storing configuration information

(D) As noted on page 9, 2nd full paragraph, "a small executable program is not a command line utility that is executable from a command line prompt".

Examiner's Response:

Examiner disagrees. A utility is a small program. Buxton disclosed a user interface to enter commands to result in modifying the registry. See comments in Examiner's responses above.

Examiner maintains the rejections of the prior Office Action.

Claim Rejections - 35 USC § 112

3. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

4. Where applicant acts as his or her own lexicographer to specifically define a term of a claim contrary to its ordinary meaning, the written description must clearly redefine the claim term and set forth the uncommon definition so as to put one reasonably skilled in the art on notice that the applicant intended to so redefine that claim term. *Process Control Corp. v. HydReclaim Corp.*, 190 F.3d 1350, 1357, 52 USPQ2d 1029, 1033 (Fed. Cir. 1999). The term "a call of a command line utility...wherein the command line utility is a utility executable from a command line prompt" in claims 1, 15, and 21 is used by the claim to mean "a utility that is capable of being executed from a command line prompt" (prompt meaning some type of visual indicator, such as a flashing '>' character in a shell terminal interface or user interface), while the accepted meaning is literally "a utility that is executed from a command line prompt". The term is indefinite because the specification does not clearly redefine the term.

Claim language recites: invoking, by an application, a call of a command line utility. Claim language, Specification, and drawings fail to provide a shell terminal interface or user interface, with a 'prompt' that awaits command line utility input from a user. Therefore, Examiner's broadest reasonable interpretation is that from within an executing application, a utility call / command is made to direct (redirect) and store output. Such an event includes storing and retrieving registry values.

As defined in Operating Systems Second Edition, by H. M. Deitel (1990), p. 574: A command line consists of a command name (i.e. the name of an executable file), followed by a list of arguments separated by blanks.

Examiner requests a clarification, and indication of support shown in the Specification, that an application invokes a call, of a command line utility, wherein the application call provides an identifier, within a shell terminal or user interface, as Applicant's amendment is suggesting.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-21 and 23-25 are rejected under 35 U.S.C. 103(a) as being unpatentable over U. S. Patent No. 6,182,279 to Buxton, in view of 5,758,154 to Qureshi.

Regarding claim 1, Buxton teaches:

A method comprising:

-invoking, by an application, a call of a command line utility, the application providing an identifier in the call of the command line utility, wherein the command line utility is a utility executable from a command line prompt;

Buxton inherently invokes a utility (system level services / registry editor, col. 8, line 7) to modify and store the customized components created. An identifier is inherently provided to register the customized component in the registry. Col. 7, line 65-col. 8, line 10, "Container may comprise any stand alone application capable of embedding OLE controls. A container **interacts with the WIN32 APIs** through the OLE libraries **in order to insert** OLE objects or controls **into the operating system registry**...The OLE libraries function to call the WIN32 APIs to locate (using a type of identifier) registered objects in registry and to insert and create object dialog (utility calls identify objects inserted / created / modified in the registry)". (emphasis added)

Official Notice is taken that an executable command to modify the registry, is capable of being executed from a command line prompt.

-receiving output from the command line utility;

As an example, a utility modifies (utility output) the registry (col. 8, lines 8-11).

-storing the command line utility output in a system storage at a location identified by the identifier;

As an example, the information related to the modification of a component (utility output) is stored at a registry key (a location identified by the identifier). Also, see col. 14, lines 20-28.

-retrieving, by the application, the command line utility output from the system storage at the location identified by the identifier.
As an example, the OLE libraries use the registry key information (retrieve output from system storage at identifier location / registry key) to find information about the OLE control. (col. 10, lines 8-10)

Buxton suggested receiving commands via command line, which results in modifying the registry (system storage) and storage. Buxton failed to specifically disclose a "command line utility". Buxton suggests that the command line input (an object that consists of modifications to base component) is directed (DIR utility – a command utility) to storage, and the registry is edited (a command utility), but did not explicitly disclose 'command line utility'.

Qureshi clearly discloses (col. 3, lines 26-46) an application call to a registration routine ('command line utility') via system calls, to store an identifier in the registry.
Col. 4, lines 2-7, "When a computer program invokes (commands) the registration routine, it passes to the registration routine a configuration file that contain a description of the configuration information of the computer program. The registration routine open the configuration file and add the configuration information to the registry." Similar to Buxton's invention, Qureshi disclosed (col. 4, lines 58-67, "Moreover, when a new component is added to

an already installed application, the user previously was required to re-execute the entire installation procedure. With the registration DllRegisterServer routine, a new SRG file that includes only the configuration information required by the new component can be distributed along with the new component, and the **registry can be properly updated by simply calling DllRegisterServer (a command line utility call) and supplying as an argument the pathname of the SRG file containing the configuration information required by the new component**." (emphasis added) Col. 5, lines 56-59, "The application programs invoke operating system routines (invoking by an application, a call of a command line utility) in order to access the various services provided by the operating system, including routines for reading and writing files. The **application programs invoke the registration...routines** provided by the registration DLL to register...configuration information, passing an indication SRG file that contains the configuration information to register (**identifier**)...**The registration...routines** of the registration DLL **call operating system routines (utilities) to write information to the registry** file..." (emphasis added) Col. 8, lines 48-67, "DllRegisterServer opens the specified key...calling the system routine RegOpenKey...DllRegisterServer creates the key in the registry...DllRegisterServer then calls the subroutine Process_key_values...The subroutine...returns TRUE...and returns FALSE..."

The following dictionary definitions further support the rejection:

As defined in Microsoft Computer Dictionary, 5th Edition, page 111, **command line**: A string of text written in the command language and passed to the command interpreter for execution. (emphasis added) As defined in Microsoft Computer Dictionary, 5th Edition, page 544, **utility**:

A program designed to perform a particular function; the term usually refers to software that solves narrowly focused problems or those related to computer system management. (emphasis added)

Regarding the Applicant's use of 'command line utilities', see Specification, page 1, lines 8-15. "Often, such system information is available only through command line (e.g., console) utilities. That is, utilities that are accessible only through a command line interface. Illustrative command line utilities include 'dir' and 'net view' commands available in the Microsoft WINDOWS operating system..." Page 2, lines 20-24, "In one illustrative embodiment, command line utility output is stored in a system registry database...In another illustrative embodiment, command line utility output is stored in a shared system memory." Page 4, line 10 recites an example of an application that invokes a system call. Page 5, lines 6-9, Applicant states, "It will be recognized that the registry is an operating system generated and maintained database which application programs, application setup programs, and the operating system itself use to store configuration information." Page 5, line 19-page 6, line 2, Applicant admits that various system calls (system utilities) defined in the registry application programming interface are used to add / delete / modify / store in the registry.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to have modified Buxton's GUI to include specific details related to system utilities / command line utilities for effectively accessing and/or interacting with system storage as suggested by Buxton (col. 8, lines 45-47). Furthermore, specific details, as disclosed by Qureshi,

commanding the modification of system storage (the registry), using such executable routines included in the registration DLL, redirecting the received output to be stored at a location identified by the identifier (registry keys and sub-keys), provide defined operating system commands used to provide options to a Windows language programmer for customizing the registry as needed for initialization, enhancing accessibility when a graphical user interface is not desired to support an executing program. WIN32 system utilities are well known in the art and reflect knowledge of one of ordinary skill in the art at the time of the invention.

Regarding claim 2, Buxton teaches:

-providing the identifier comprises providing an identifier that identifies one or more entries in a system registry database.

(Fig. 2, item 205 and col. 13, lines 14-15, "...registry keys are created..." Also see col. 14, lines 29-59, "To facilitate loading of template onto another system...a number of registration key or subkey are included with template. Each template may have the keys 450A-I, as illustrated in Fig. 4C...Key 450H contains information indicating the name of the storage object in template storage file where initialization data...may be located...Key 450I contains information identifying the CLSID...)

Regarding claim 3, Buxton teaches:

-providing a root key identifier.

(Col. 11, line 2: "Most OLE object application information is stored in subkeys under the CSLID root key..." Also see col. 17, lines 35-41, "Component loader loads, verifies and checks

the license of a component by replacing in registry the InProcessServer 32 entry, i.e. key 450A...and adding additional registry keys 450B-J, as previously described, that will let the component loader (receiving a root key identifier) then load the correct OLE control.")

Regarding claim 4, Buxton teaches:

-providing a sub-key identifier.

(Col. 11, line 2 and col. 14, line 31: To facilitate loading of template...a number of registration or subkey are included with template...")

Regarding claim 5, Buxton teaches:

-system registry database comprises an operating system registry database.

(Col. 4, line 49: "Operation of computer system is generally controlled...by operating system software, such as...Windows95...")

Regarding claim 6, Buxton teaches:

-providing a system storage identifier.

(Col. 12, lines 20-21, "...users identify...templates to be packaged..." Also for another example of receiving a system storage identifier, see col. 20, lines 42-45, "...relevant character string from the registry is converted to CLSID. The component loader (receives a system storage identifier) then calls the GetClassObject to retrieve the real component's class factory...")

Regarding claim 7, Buxton teaches:

-providing the system storage identifier comprises providing an identifier indicating a system registry.

(Col. 10, line 66 – col. 11, line 4: A CLSID identifies the functionality of an object class that can display...access to property values...A subkey is used by an OLE to find out information about the control.")

Regarding claim 8, Buxton teaches:

-providing an identifier indicating shared system memory.

(Col. 8, lines 6-7: "OLE libraries (shared) comprise the set of system-level services in accordance with the OLE specification...")

Regarding claim 9, Buxton teaches:

-providing the identifier indicating shared system memory identifies a system clipboard memory.

(Col. 11, line 6: "An FORMATETC...is an OLE data structure which acts in a generalized clipboard format...")

Regarding claims 10, Buxton teaches:

-receiving output directly from the command line output utility.

As an example, a utility modifies (utility output) the registry (col. 8, lines 8-11).

Regarding claim 11, Buxton teaches:

-receiving output from the command line output utility through a subsequent command line output routine.

As an example, (col. 8, lines 28-29) "Data items within the registry are retrievable (receive output) via calls (from utility call) to the WIN32 APIs."

Regarding claim 12, Buxton teaches:

-associating each line of command line utility output with a line identifier in the system storage.

As an example, (col. 3, lines 1-9) "Template storage with a means for indexing, including key information associated with the template. "...a memory having one or more locations, means for indexing one or more locations within the memory..." Also col. 13, lines 35-44, templates are stored with an enumerated decimal number: "Each template is stored in an ISTORAGE whose name is unique...and may have the form TEMPLEnnn, where nnn may be a decimal number.")

Regarding claim 13, Buxton teaches:

-setting each line identifier to a value corresponding to a position of that line in the command line utility output.

(Rejection of claim 12 is incorporated and further claim contains limitations as recited in claim 12. Therefore claim 13 is rejected under the same rational as claim 12.)

Regarding claim 14, Buxton teaches:

-setting a default value of the provided identifier to equal the total number of command utility output lines stored in the system storage. (Rejection of claim 12 is incorporated and further

claim contains limitations as recited in claim 12. Therefore claim 14 is rejected under the same rational as claim 12.)

Regarding claim 15, Buxton teaches:

A program storage device, readable by a computer, comprising instructions stored on the program storage device for causing the computer to:

-cause an application to invoke a call of a command line utility, the application providing an identifier in the call of the command utility;

-receive output from the command line utility;

-store the command line utility output in system storage at a location identified by the identifier;

-cause the application to retrieve the command line utility output from the storage at the location identified by the identifier.

See rejection of limitations in claim 1 above. This is a "program storage device" version of claim 1. See Figure 2 regarding Buxton's disclosure of a "program storage device."

Regarding claim 16, Buxton teaches:

-instructions to store command line utility output in an operating system registry database.

As an example (Fig. 2, item 205 and col. 13, lines 14-15), "...registry keys are created..." and (col. 13, lines 10 – 15) "...Template storage DLL ensures all additional registry keys...are created..." Modified components cause the registry keys to be created / edited / modified (REGEDIT utility).

Regarding claim 17, Buxton teaches:

-instructions to store command line utility output in an operating system maintained volatile memory.

As an example, (Fig. 1, item 110-volatile storage).

Regarding claim 18, Buxton teaches:

-instructions to receive one or more lines of output from the command line utility.

See rejection of limitation in claim 1 above.

-instructions to store each of said one or more lines of output in the system storage.

As an example, (col. 14, lines 26-29) "The remainder of the operating system registry entries are generated by code (instructions to store) in the template storage DLL and are stored in both registry (store output / modified component data in system storage) and the template.")

Regarding claim 19, Buxton teaches:

-instructions to associate a unique identifier with each of the one or more lines of output stored in the system storage.

See rejection of limitations in claim 2 above.

Regarding claim 20, Buxton teaches:

-instructions to set a value associated with the received identifier in the system storage equal to the number of lines of output stored in the system storage.

(Rejection of claim 18 is incorporated and further claim contains limitations as recited in claim 12. Therefore claim 20 is rejected under the same rational as claim 12.)

Regarding claim 21, Buxton teaches:

A computer system, comprising:

-a processor;

-a command line utility;

-an application executable on the processor, the application to call the command line utility, the application to provide an identifier in the call;

-a system storage having a location identified by the identifier, the location identified by the identifier to store an output of the command line utility,

-the application to retrieve the command line utility output from the location identified by the identifier.

As an example, see FIG. 1. Claim 21 contains limitations as recited in claim 1, therefore claim 21 is rejected under the same rational as claim 1.)

Regarding claim 23, Buxton teaches:

-the command line utility comprises a first command line utility, and wherein invoking the call by the application comprises invoking a call to pipe output of a second command line utility to the first command line utility...

-wherein storing the command line utility output comprises storing the command line utility output of the first command line utility.

Chaining utilities, piping the output of a second utility as input to a first utility is known in the art. Col. 8, lines 6-7 disclose the OLE libraries comprise the set of system level services (system utilities). As an example of system utilities (col. 20, lines 17-43) Buxton disclosed reading a sub-key from the registry, use the output to determine the real component CLSID, determine whether a valid certificate and license exist, pipe the relevant character string to a CLSID, etc.

Regarding claim 24, Buxton teaches:

-the command line utility comprises a first command line utility, and wherein invoking the call by the application comprises invoking a call to pipe output of a second command line utility to the first command line utility...

-wherein storing the command line utility output comprises storing the command line utility output of the first command line utility.

This is a 'program storage device' version of claim 23 above. See rejection of claim limitations in claims 15 and 23 above.

Regarding claim 25, Buxton teaches:

-the command line utility comprises a first command line utility, the system further comprising a second command line utility, the application to invoke a call that causes output of the second command line utility to be piped to the first command line utility...

-the location identified by the identifier to store output of the first command line utility.

This is a 'system' version of claim 23 above. See rejection of claim limitations in claims 21 and 23 above.

Conclusion

7. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached at (571) 272-3708. The fax phone number for the organization where this application or proceeding is assigned: 571-273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman

03/07/2007

Primary Examiner